Fast Exact and Heuristic Methods for Role Minimization Problems

Alina Ene[*]
Princeton University
aene@princeton.edu

William Horne[†]
Hewlett-Packard
william.horne@hp.com

Nikola Milosavljevic[‡]
Stanford University
nikolam@stanford.edu

Prasad Rao[†]
Hewlett-Packard
prasad.rao@hp.com

Robert Schreiber[§]
Hewlett-Packard
rob.schreiber@hp.com

Robert E. Tarjan[*, †]
Hewlett-Packard and
Princeton University
robert.tarjan@hp.com

ABSTRACT

We describe several new bottom-up approaches to problems in role engineering for Role-Based Access Control (RBAC). The salient problems are all NP-complete, even to approximate, yet we find that in instances that arise in practice, these problems can be solved in minutes. We first consider role minimization, the process of finding a smallest collection of roles that can be used to implement a pre-existing user-to-permission relation. We introduce fast graph reductions that allow recovery of the solution from the solution to a problem on a smaller input graph. For our test cases, these reductions either solve the problem, or reduce the problem enough that we find the optimum solution with a (worst-case) exponential method. We introduce lower bounds that are sharp for seven of nine test cases and are within 3.4% on the other two. We introduce and test a new polynomial-time approximation that on average yields 2% more roles than the optimum. We next consider the related problem of minimizing the number of connections between roles and users or permissions, and we develop effective heuristic methods for this problem as well. Finally, we propose methods for several related problems.

1. BOTTOM-UP ROLE MINING

Access control allows organizations to grant or deny access of something to someone. Traditionally, access controls have been maintained locally with the entity to which access is being granted. This causes significant management overhead in large organizations where the access controls are spread across the enterprise, and it is very difficult to rationalize why certain users have access to certain things while others do not. In large organizations there is a significant security concern that individuals might have access to things they do not need to do their jobs, thus creating a risk of fraudulent or otherwise damaging actions.

Role based access control (RBAC) attempts to remedy the situation by centralizing access controls and aligning them with business needs. Here, users are assigned to a role, such as "accounts receivable clerk", and that role in turn is defined as the set of permissions such a person needs to do his or her job. RBAC can be centralized across multiple applications, which leads to an easier management task, and in principle roles can be defined to minimize the possibility that an employee is given permissions beyond the scope of the job.

The first step in setting up any kind of RBAC system is to define a set of roles; this is a step in a process known as role engineering. The specification of roles by fiat (based, presumably, on business acumen) is referred to as top-down role engineering [3]. This process is labor intensive and therefore costly [5]. Analysis of the existing access controls can be done to derive a set of roles which are inherently defined in an organization. This type of bottom-up analysis is called role mining or role discovery. The top-down and bottom-up approaches are not incompatible: the roles discovered bottom-up can then be modified and approved top-down; a partial set of roles specified top-down can be augmented and completed by bottom-up approaches. In this paper, we consider the bottom-up approach.

Thus, the task in bottom-up role engineering is to find a set R of roles, together with a set of user-to-role assignments E_{UR} and a set of role-to-permission assignments E_{RP} such that user u has permission p if and only if (iff) there is a role $r \in R$ such that both (u, r) is part of the user-to-role assignment relation E_{UR} and (r, p) is in the role-to-permission relation E_{RP}. The Role Mining Problem (RMP) is to find a smallest set of roles for which this is possible.

[*]Department of Computer Science, Princeton University, 35 Olden Street, Princeton, NJ 08540-5233

[†]HP Labs, 5 Vaughn Drive, Suite 301, Princeton, NJ 08540 USA

[‡]Department of Computer Science, Stanford University, 318 Campus Drive, Stanford, CA 94305

[§]HP Labs, 1501 Page Mill Rd., Palo Alto, CA 94304-1126 USA

SACMAT Estes Park, Colorado, June 2008

1.1 Our contributions

Our contributions to the RMP are as follows. We introduce (Sect. 4.1) a method to find the best possible solution to the RMP that succeeds for each of a collection of practical problems that we have used as a test suite. The method uses fast graph reductions that allow recovery of the optimum solution from the solution to a problem on a graph smaller than the original input. For most of the realistic problems for which we have tried them, these reductions completely solve the problem by themselves. On the remaining problems, they reduce the problem enough that we can find the exact solution with an exponential (in the worst case) method.

We introduce (Sect. 4.2.1 and 4.2.2) and test a fast approximation method that is within 12% of best possible on all of our test cases, and 2% on average across all the test cases. We give a method for computing (Sect. 4.3.1) a lower bound that is sharp for seven of nine test cases and are within 3.4% on the other two, and is fast to compute. Finally we show experimentally (Sect. 4.3.3) that the matrix rank of a bipartite graph adjacency matrix approximates quite closely the number of roles in a solution to the RMP.

A second problem that we attack is the minimization not of $|R|$ but rather some measure of the total size of the tripartite representation. Indeed, Zhang, Ramamohanarao, and Ebringer [23] proposed $|R| + |E_{UR}| + |E_{RP}|$ as a measure to be minimized, and a heuristic for reducing it. We consider this problem and their approach in Section 5. We have tried a variant of their approach on large problems arising in practice and found that, for some cases, a significant reduction in the number of edges is possible with only a small increase in the number of roles.

Finally, we propose (Sect. 6) a way to generalize any approach to the RMP in order to implement a system in which users are assigned to groups of users (possibly overlapping, so a user may be part of several groups), which are granted roles, which confer permissions. We find that a convergent iterative process that alternately updates and improves the groups, then the roles, then the groups, *etc.*, is effective.

2. PROBLEMS ADDRESSED

As our methods and our explanations are couched in the language of graph theory, we review some terminology before we get started. We then define the problems addressed in this paper.

2.1 Graphs

In a graph $G = (V, E)$, with vertex set $V = V(G)$ and edge set $E = E(G)$, we say that $v \in V$ and $w \in V$ are *adjacent* if $(v, w) \in E$. The vertices v and w are called the *endpoints* of the edge (v, w). E is a set of unordered pairs; the edges of G are undirected. For a vertex v, we denote by $\Gamma(v)$ the set $\{w \mid (v, w) \in E\}$ of its neighbors. If S is a subset of the vertices $V(G)$, then the subgraph *induced* by S is the graph whose vertex set is S and whose edges are the members of $E(G)$ whose two endpoints are both in S. If S is a set of vertices, we denote by $G[S]$ the subgraph of G induced by S. Similarly, for any set T of edges, we denote by $G[T]$ the subgraph of G induced by the endpoints of the edges in T. Thus $T \subseteq E(G[T])$. For $v \in V$, the *star* centered at v is $G[v \cup \Gamma(v)]$.

A *clique* in a graph is a set of pairwise adjacent vertices. An *independent set* in a graph is a set of vertices, no two of which are adjacent. A graph G is said to be *bipartite* if $V(G)$ can be partitioned into two subsets V_1 and V_2 such that for every edge $(v_1, v_2) \in E(G)$, $v_1 \in V_1$ and $v_2 \in V_2$. A *biclique* in a bipartite graph is a set of vertices $C_1 \subseteq V_1$ and $C_2 \subseteq V_2$ such that $(c_1, c_2) \in E$ for all $c_1 \in C_1$ and $c_2 \in C_2$. An independent set, clique, or biclique is said to be *maximal* if it is not a proper subset of a larger independent set, clique, or biclique, respectively.

A collection $\mathcal{C}$ of bicliques is a *biclique cover* (of the edges) of G if for every edge (u, v) of G there is a biclique $B \in \mathcal{C}$ such that $u \in B$ and $v \in B$; equivalently, $(u, v) \in E(G[B])$. We say that B *covers* (u, v) in this case. A minimum cardinality collection of bicliques that covers the edges of a given bipartite graph is a minimum biclique cover (MBC). The size of an MBC is known as the *biclique cover number*, denoted $\mathrm{bc}(G)$.

2.2 Problems

To begin, we are given an undirected bipartite graph $G = (V, E)$ in which the vertex set V has been partitioned into disjoint subsets U and P, where the members of U are the users and the members of P are the permissions, and where E is a set of pairs (u, p) in which $u \in U$ and $p \in P$. The pair $(u, p) \in E$ iff user u is granted permission p.

Our work addresses precise formulations of several role engineering problems that were introduced by Vaidya, Atluri, and Guo [21] and by Vaidya, Atluri, and Guo [21] at SACMAT 2007. The first problem we attack was called the Role Minimization Problem (RMP) by Vaidya; it is to find a minimum cardinality set of roles that can be used to implement a given user-permission relation. Thus, the task is to find a smallest possible set R of roles, together with a set of user-to-role assignments E_{UR} and a set of role-to-permission assignments E_{RP}, such that

$$E = \{(u, p) \mid \exists r \in R,\ (u, r) \in E_{UR},\ (r, p) \in E_{RP}\}. \qquad (1)$$

In other words, our original bipartite graph G has an edge from u to p iff there is at least one path of length two from u to p in the tripartite graph

$$G_{RB} = (U \cup R \cup P, E_{UR} \cup E_{RP})\,.$$

Given a set of roles and pair of relations E_{UR} and E_{RP} satisfying (1), each role is adjacent to a subset of users and a subset of permissions, and each of these users has each of these permissions. The users and permissions connected to a role therefore induce a biclique in G. The set $\mathcal{C}$ of subgraphs induced by $\{u \mid (u, r) \in E_{UR}\} \cup \{u \mid (r, p) \in E_{UR}\}$ as r varies over all of R is a biclique cover of G. Similarly, given a biclique cover of G, we obtain a set R of roles (identified with the bicliques) and a pair relations E_{UR} and E_{RP} satisfying (1). The role mining problem is therefore equivalent to the problem of finding an MBC. This is an NP-hard problem [14] that is known to be hard to approximate. Results of Simon [19] and of Lund and Yannikakis [11] show that MBC has no polynomial time approximation with factor n^{δ}

for $\delta > 0$ unless $P = NP$.[1] To date, heuristics have been proposed, but they remain largely untested on practical examples.

There are a variety of equivalent ways to view the RMP. One may view it as factorization of a binary relation into a join of two relations, factorization of a $\{0, 1\}$ matrix as a boolean product [18], tiling of a database [6], or the representation of a given bipartite graph as the transitive closure of a tripartite graph in which the set of roles is a newly introduced subset of vertices. We prefer the graph point of view, but all of our approaches are applicable no matter which of these fully equivalent viewpoints is taken. Cornaz and Fonlupt discuss several practical applications of the MBC problem [2].

2.2.1 *Reduction to minimum clique partition and chromatic number*

A minimum clique partition (MCP) of a graph is a smallest collection of cliques such that each vertex is a member of exactly one of the cliques; it is a partition of the vertices into cliques. The graph coloring problem is to find a smallest partition of the vertices of a graph into independent sets. The chromatic number of a graph is the number of independent sets in a smallest partition of the vertices into independent sets. Here, we introduce a reduction of the MBC problem to the MCP and the graph coloring problems that will be useful to us in building and explaining algorithms.

Let $G = (V, E)$ be a given bipartite graph in which we seek a minimum biclique cover. We construct a new, undirected, unipartite graph $G' = G'(G)$, which is an edge dual of the given bipartite graph G, as follows. The edges of G become the vertices of G'. A pair of vertices in G' are connected by an edge iff the endpoints of the corresponding edges of G induce a biclique of G; see Figure 1. Thus,

$$G' = (E, \{(e_1, e_2) \mid e_1 \text{ and } e_2 \text{ induce a biclique of } G\}).$$

If two edges can be covered by a single biclique, then clearly the (three or four) vertices that are endpoints of these two edges are completely connected and form a small biclique, so the two edges must be adjacent; hence, a pair of non-adjacent edges cannot both be covered by a single biclique.

The vertices of a (maximal) clique in G' correspond to a set of edges of G whose endpoints are a (maximal) biclique in G. The edges covered by a (maximal) biclique of G induce a (maximal) clique in G'. Thus, every biclique edge cover of G corresponds to a collection of cliques of G' whose union contains all of the vertices of G'. From such a collection, a clique partition of G' can be obtained by removing any redundantly covered vertex from all but one of the cliques to which it belongs. Similarly, any clique partition of G' corresponds to a biclique cover of G. Thus, the biclique cover number of a bipartite graph G is equal to the clique cover number (the size of a minimum clique partition) of $G'(G)$.

A clique partition of a graph $G = (V, E)$ is a coloring of its

[1]One of us (A. Ene) provides a more complete picture of the theory of the difficulty of the problem in an unpublished report [4].

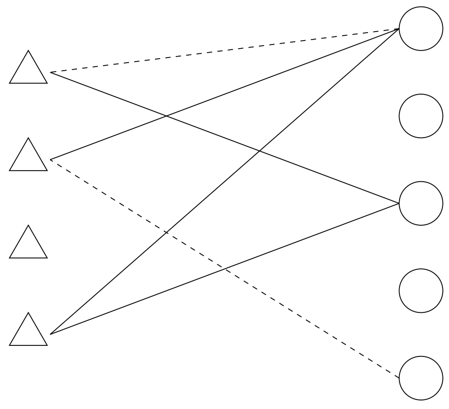


Figure 1: A pair of edges is adjacent if their endpoints form a biclique. The dashed edges are not adjacent: a cross edge from the upper triangular vertex to the lower circular vertex is missing. They cannot both be covered by a single biclique.

complement, $\widehat{G} = (V, (V \times V) \setminus E)$. Thus, the biclique cover number of a bipartite graph G is the chromatic number of $\widehat{G'(G)}$.

3. RELATED WORK

Top-down approaches and assisted top-down approaches are discussed in several papers. Kuhlmann *et al.* [8] use clustering and data mining techniques to obtain structural and statistical information that can be used to construct roles. Schlegelmilch and Steffens [17] take a hierarchical clustering approach to identify role hierarchies, and describe a tool allowing users to visualize and manage role hierarchies.

In a 2006 paper, Vaidya, Atluri, and Warner [22] proposed a bottom-up method that first enumerates a set of candidate roles (bicliques) and then selects from among these. The candidates are enumerated by considering each user, the permissions of that user, then all users that have this set of permissions. Then all intersections of the permission sets of this initial set of bicliques are generated and added to the candidate role set. Finally, this (potentially large) set of candidates is scanned in some priority order to select the cover. Tests were done, but only on synthetic datasets. Subset enumeration techniques had been advocated earlier [16].

In their 2007 paper, Vaidya, Atluri, and Guo [21] formally define the RMP, in effect as the minimum biclique cover problem. They advocate a greedy heuristic that grows a cover by including, at each step, the biclique that covers the largest possible set of previously uncovered edges. Unfortunately, this step requires the solution of the maximum edge biclique problem, which is itself NP-hard [15], so they advocate a branch and bound method. They present no test of its complexity or efficacy in practice.

4. ROLE MINING VIA BICLIQUE COVERS

We consider the Role Minimization Problem; that is, we give methods for finding a biclique cover of the edges of a bipartite graph. We give a practical and exact (but exponential in the worst case) algorithm and a fast (polynomial), accurate approximation algorithm. In a discussion section, we give a lower bound algorithm for the biclique cover number; we then show how to find the best biclique cover in which each biclique is a star; and we show how the rank of the adjacency matrix provides an accurate estimate of biclique cover number.

4.1 Exact upper bounds via reduction

In this section we consider techniques that allow us to find the exact solution to the MBC problem even for very large instances. Recall that the MBC problem for a given bipartite graph G is equivalent to the MCP problem for G's edge dual G', as explained in Section 2.2.1. Our plan, then, is:

1. Construct the dual $G'(G)$.
2. Find a minimum clique partition of $G'(G)$ *via* graph reduction:
   (a) Remove vertices and their incident edges from G' according to the strategy below, until no further removal is possible.
   (b) Form the complement of the resulting reduced graph, the *irreducible kernel*.
   (c) Use a branch and bound backtracking method to color this graph.
   (d) Each color class is a clique in a clique partition of the irreducible kernel.
   (e) Include the removed vertices in the reverse of the order they were removed, recovering a clique partition of G'.
3. Each clique of the MCP is a set of edges of G. Construct the set of endpoints of these edges. It induces a biclique of G. Create one role corresponding to this biclique. The set of roles thus created solves the RMP.

We propose to construct the dual graph G' by a doubly nested loop over edges, finding all the pairs of edges that are adjacent, at cost $O(|E(G)|^2)$, to get the edge set of the dual graph G'. We then find an MCP $\mathcal{C}$ of G' via the reduction strategy below.

The basic idea of the reduction method is best explained as a reduction for the MCP problem. For the remainder of this section, $G = (V, E)$ is a simple graph (not in general bipartite), and we seek a (smallest) set $\mathcal{C}$ of cliques of G that partitions the vertices V: each $v \in V$ is a member of one of the members of $\mathcal{C}$.

We say that a vertex d is the dominator of a vertex g if g and all of its neighbors are a subset of d and its neighbors:

$$d \text{ dominates } g \;\Leftrightarrow (\{\, g\,\} \cup \Gamma(g)) \subseteq (\{\, d\,\} \cup \Gamma(d)) \;.$$

The reduction strategy for MCP is as follows. If a vertex of $x \in V(G)$ has no neighbors, then there is an MCP of G that consists of a clique containing x alone, together with an MCP of the subgraph induced by $V \setminus \{x\}$. If the vertex d dominates the vertex g, then there is an MCP of G that consists of an MCP of the subgraph induced by $V \setminus \{d\}$, modified to include d in the unique clique that contains g. We thus have a recursive strategy: find an isolated vertex v, if any, and append it (a single-vertex clique) to an MCP (found by recursion) of $G[V\setminus\{v\}]$; alternatively, find a vertex d that dominates a vertex g, construct (recursively) an MCP of $G[V \setminus \{d\}]$, and add d to the (unique) clique that contains g.

Lemma 1. *This recursive strategy constructs an MCP of G.*

Proof. If G has an isolated vertex x, then the only clique that can cover x is $\{x\}$. Thus, any clique partition, and therefore every MCP, consists of $\{x\}\cup\mathcal{C}$ where $\mathcal{C}$ is an MCP of $G[V \setminus x]$. This is exactly what the strategy provides in this case. Note that we have shown that $\mid MCP(G) \mid = 1+ \mid MCP(G[V \setminus x]) \mid$ in this case.

In the other case, we have to show that we construct a set of vertex disjoint cliques of G that partitions the vertices $V(G)$, and that there is no smaller clique partition. The recursive call produces a set of cliques of $G[V \setminus d]$ that partitions all vertices save the removed dominator vertex d. A clique of $G[V \setminus v]$ is a clique of G for any vertex v, so these are (disjoint) cliques of G. Then d is added to the clique that contains the dominated vertex g, and so is covered; the collection $\mathcal{C}$ so modified therefore partitions V. And the clique C to which we add d remains a clique. Because C was a clique that contains g, we have that $C \subseteq (\;\{\,g\,\} \cup \Gamma(g)\;)$, and since d dominates g, $C \subseteq \Gamma(d)$. In other words, every member of C is a neighbor of d. Hence, $C \cup \{\,d\,\}$ is a clique in G. This shows that we have built a clique partition. It must be an MCP of G. If not, there is another of smaller cardinality. Removal of a vertex from a clique yields a clique. We can remove d from the one member of this smaller clique partition of G to which it belongs, and the result is a smaller clique partition of $G[V \setminus d]$ than the MCP found by the recursive call, a contradiction. $\square$

Our implementation follows this plan conceptually, but not in detail. In particular, we do not construct G', because its edge set can be as large as $|E|^2$. Rather we find the dominators in G' by working with the data structure representing G.

We implement the algorithm as an iterative process, reducing the number of edges to be covered until an irreducible kernel – that is a graph with neither isolated vertices or dominators – is obtained. The reduction by removal of an edge is accomplished by a marking of edges as having been removed, rather than their actual removal; we must retain all of G, as it is our implicit representation of G'.

When no further reduction is possible, we generate the irreducible kernel of G', recast the MCP problem for the irreducible kernel into the corresponding coloring problem for its edge-complement, and use well known techniques to find a coloring [12]. It is then a simple process to recover the minimum biclique cover.

The complexity of the reduction is $O(|E|^3|V| \log |V|)$. There are at most $|E|$ iterative steps, and at each step there is a search over (at most) all edge pairs to find a dominator. The check to see whether or not one edge dominates another, which is a check on whether one set of vertices is a subset of another, costs $|V| \log |V|$. (The check is performed by sorting the indices then comparing them; if |V| storage cells are available, the check can be done in time O(|V|).)

The experiments below show that this approach is very powerful in practice. It solves seven of our nine test cases by reduction alone. For the other two, we are able to find best-possible colorings of the very small irreducible kernels.

4.2 Heuristics

Although the algorithm discussed in the previous section can be used to solve large problems in practice, this technique has limits. As a practical matter, there will be bipartite graphs for which finding a minimum biclique cover is not feasible in a reasonable amount of time. In this section we discuss fast heuristic strategies that produce very good results, as will be seen when we compare them with optimum solutions and lower bounds.

4.2.1 A greedy algorithm

We propose a greedy algorithm that builds a biclique cover by identifying and including one biclique at a time in the cover until all edges are covered. We construct bicliques as follows. Take a vertex v according to some criterion – we discuss several of these below. Find its neighbors, $\Gamma(v)$, and then find the set $\Xi(\Gamma(v))$ of vertices that are adjacent to all of $\Gamma(v)$. (If v is a user, we find his or her permissions and then find all other users who also have all of these permissions; if on the other hand v is a permission, we find all users that have this permission and then all permissions that all of these users have.) Clearly $v \in \Xi(\Gamma(V))$, so $\Xi(\Gamma(V))$ is nonempty and the vertex subset $\Xi(\Gamma(V)) \cup \Gamma(v)$ is a biclique. (Rymon [16] also uses this technique to build bicliques.)

Vaidya, Atluri, and Guo [21] also take the "add one biclique at a time" approach. But they advocate choosing at each step a biclique that covers as many uncovered edges as possible. Unfortunately, the subproblem of finding this next biclique is itself NP-hard [15]. It would be interesting to pursue fast methods for finding bicliques that cover many, if not the most, uncovered edges.

We have experimented with our method. We tried the following criteria for selecting the next vertex v as the seed for $\Xi(\Gamma(v))$: a vertex with fewest uncovered incident edges, with most uncovered incident edges, or with any number other than zero of uncovered incident edges. Selecting a node with the fewest (but not zero) uncovered incident edges tends to give a slightly better result than selecting one with the most. Selecting a random vertex does not do as well as either of these strategies. The method is fast enough that one can use both (fewest / most) of the competitive variants and simply select the better final result, which is what we do in our tests.

4.2.2 Lattice-based postprocessing

In this section we describe a procedure that we use to reduce the number of roles found by a first heuristic, such as the greedy algorithm of the previous section. The combination of these two methods derives sets of roles that come close to best possible in practice.

Our procedure employs a technique of Zhang *et al.* [23], who used it for reducing edge count as described in the following section. Let a tripartite graph G_{RB} (whose middle vertices are interpreted as roles) be given. For the remainder of this section, we identify the role $r \in R$ with the subset $\mathrm{Perms}(r) \in P$ that this role confers and to which r is adjacent in G_{RB}. Viewing each role thus, as the set of permissions to which it is adjacent, consider the lattice of roles with respect to the subset relation. We write $r_s \subset r_S$ if $r_s \subseteq r_S$ and $r_s \neq r_S$. Let r_S be a role that contains other roles, and consider the part of r_S that is not contained in any other role: $r'_S \equiv r_S \setminus \cup_{r_s \subset r_S} r$. We can grant the same permissions to a user with role r_S by assigning him or her the roles r'_S and each maximal $r_s \subset r_S$. If r'_S is empty, the number of roles is thereby reduced.

For each iteration of the algorithm we replace r_S with r'_S for every role in the lattice. This yields a new set of roles. We continue until the lattice is completely flat: no role covers any other role.

Comparison of the results obtained this way with the MBCs found by the reduction method shows that this approach is extraordinarily effective in practice.

4.3 Discussion

We cover some related topics in this section: a lower bound, finding the best star cover, and approximating boolean rank by real rank.

4.3.1 A lower bound on roles

To assess how well heuristics perform, we need a lower bound on the number of bicliques in an exact cover.

The size of an independent set in a graph is clearly a lower bound on its clique cover number. The biclique cover number $\mathrm{bc}(G)$ is the clique cover number of its dual $G'(G)$. We therefore get a lower bound on $\mathrm{bc}(G)$ by finding a maximal independent set I of edges, that is, a maximal independent set in the graph G'.

Finding a maximum set of pairwise independent edges in a bipartite graph is NP-complete [13]. We have therefore used a greedy heuristic to find maximal independent sets. Starting with an empty set of edges, we repeatedly choose an edge to add to the independent set. As a heuristic, we choose an edge with the fewest adjacent edges, breaking ties at random. We then remove that edge and the edges adjacent to it from the graph. The process is repeated until no edges are left. We run the randomized algorithm a number of times in order to increase the lower bound achieved. On the nine examples on which we test it (see Section 7.2), the bound is exact in seven cases, off by 2% in one, and off by 3.5% in the last.

4.3.2 Maximum matchings and biclique covers

A matching in a bipartite graph (V, E) is an edge subset $S \subseteq E$ with the property that no two members of S have a common endpoint. A matching is maximal if every edge not in the matching has an endpoint in common with an edge that is in the matching. A matching of largest cardinality is a maximum matching. A maximum bipartite matching can be found in $O(|V|^{2.5})$ time [7].

A star is a subgraph induced by a single vertex (the center) and its neighbors. There are trivial edge covers consisting of stars: the stars centered at the members of U and the stars centered at the members of P. These are not competitive with covers found by our heuristic and our exact methods. What about the best possible star cover? It is easy to see that the size of the smallest star cover is the size of a smallest set of vertices incident on every edge (called a minimum vertex cover). According to König's Theorem [1], this is equal to the size of a maximum matching. Thus, we can quickly determine the size of a best star cover; we report these results in Section 7.2 below. These too are not competitive with covers found by the exact and heuristic methods.

4.3.3 *Boolean and real rank*

Let A be a real $m \times n$ matrix. The *real rank* of A, denoted rank(A), is the smallest integer r such that A has a factorization $A = BC$ where B has r columns. If the elements of A are in $\{0, 1\}$, and if we consider the boolean product of matrices, so that the (i, j) element of the product BC is given by

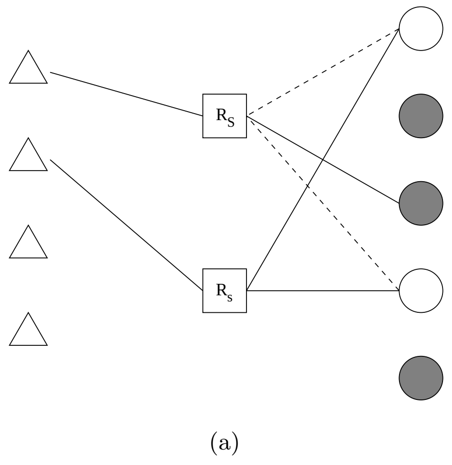
$$(BC)_{ij} \equiv \bigvee_{k=1}^{r} b_{ir} \wedge c_{rj}$$

(here $\wedge$ is multiplication and $\vee$ is boolean addition, where $1 + 1 = 1$) then the *boolean rank* of A is the smallest r such that A can be factored as above into a product BC where B has r columns. Now let $A = A(G)$ be the incidence matrix of the bipartite graph G: $A_{up} = 1$ if $(u, p) \in E(G)$ and $A_{up} = 0$ otherwise. Then the biclique cover number bc(G) is equal to the boolean rank of $A(G)$.

The nonnegative-integer rank of a matrix is the smallest r for which it can be factored as a product BC of nonnegative integer matrices in which B has r columns. It is known that the boolean rank is not larger than the nonnegative integer rank [18], but computing nonnegative integer rank is NP-complete. On the other hand, real rank can be computed fast in practice using numerical techniques. It is not the case in general that real rank bounds boolean rank either above or below. We have, however, found for our test cases that the two are remarkably close; see Table 1. Although this is not always true, we find that in all these test cases, rank($A(G)$) $\geq$ bc(G). On average it is 2.5% higher than bc(G). The practical import of these findings is not clear, but we feel that they are intriguing enough to mention.

5. EDGE CONCENTRATION

Rather than minimizing the number of roles, we may wish to minimize the size of the role-based data structure. One way to measure this size is by the sum over all roles of the number of users and number of permissions in that role; equivalently, it is the number of edges in the tripartite graph G_{RB}. This problem, known in the literature as *edge concentration* (EC), has also been shown to be NP-hard [10], and is hard to approximate as well: EC has no polynomial time approximation with factor n^{δ} for $\delta > 0$ unless $P = NP$ [4].



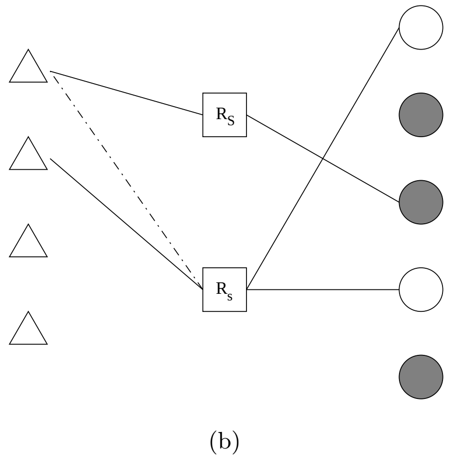


Figure 2: (a) The permissions of the role r_s are a subset of those of the role r_S. (b) The dashed edges are removed, and the dash-dotted edge is added, for a net reduction of one edge.

More generally, we could try to minimize a linear combination $\lambda_R|R| + \lambda_E(|E_{UR}| + |E_{RP})|$ of the number of roles and the number of edges in G_{RB}.

The only lower bound on the number of assignments that we know of is $|U| + |P|$, the number of vertices in G, since every nonisolated vertex must have at least one incident edge.

5.1 A heuristic for edge concentration

We do not have generalizations of the reduction method of Section 4.1 for edge concentration; this is an avenue for further research. We concentrate here on heuristic approaches.

Zhang *et al.* [23] make the following observations:

- When the permissions of one role, r_s, are a proper subset of those of a second role, r_S, one may remove the permission of the subset from the superset, thus removing edges between roles and permissions, at the expense of adding an edge from any user that has role r_S in order to also give her role r_s. This strategy may be used to reduce edges without increasing roles. The

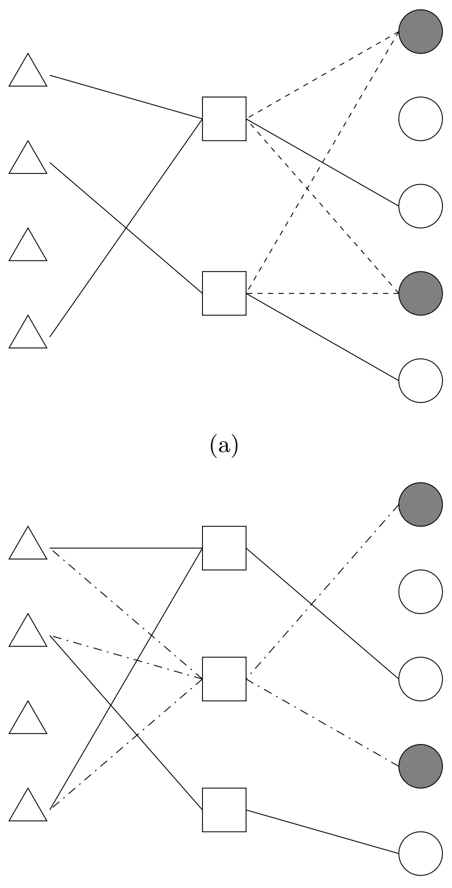


Figure 3: (a) The two roles have two permissions (gray) in common. (b) A new role (the middle one) is created and connected to the gray permissions, which are no longer part of the original two roles. Dashed edges are removed. Users who had either of the original roles now have the new role as well. The added edges are shown by black dash-dotted lines.

situation is symmetric; we can also exploit cases in which the users of one role are a subset of the users of another. A case in which this reduces edges is shown in Figure 2. As this transformation reduces edges without increasing roles, it ought to be performed as a matter of course as a postprocessing step in any role minimizer or any solver of the MBC problem for that matter, and we use it in the implementation that produced the results described in Section 7.2. Note that in the lattice-based postprocessing technique of Section 4.2.2 we apply this same graph transformation. Since our goal there is to minimize roles, we apply it without regard to the effect it has on edge count, and we do so only when r_s is a maximal proper subset of r_S, that is only when there are no intervening roles in the lattice.

- When there is substantial overlap between the permissions of a pair of roles, we can potentially remove edges by introducing a new role that has as its permissions the intersection of the permissions of these two roles, and removing the intersection from their permission sets. This is true of users as well. Figure 3 illustrates this. (In the case shown, the transformation increases the number of edges, so we wouldn't make it.)

We exploit these observations, employing a heuristic strategy to approximately minimize (a linear combination of) the number of roles and edges. We start by using one of the role minimizing algorithms of either Section 4.1 or Section 4.2.1. We then greedily improve the objective function using the transformations above, until no further improvement is possible. To do so, we find the amount of overlap among the user and permission sets of the current set of roles, and for each pair of roles whose user or permission sets have nonempty intersection, we evaluate the improvement in the objective achievable by the transformations above. We choose at each step to make a change that achieves the best possible reduction, breaking ties at random, and stopping at a local minimum (when no change of this kind yields a further reduction.) The results of Section 7.2 show that large reductions in edge count can be achieved in some cases.

6. ADDING GROUPS TO THE ROLE HIERARCHY

In a hierarchical role-based system, roles may imply other roles, so that the strict three-tier structure of users, roles, and permissions is generalized. In one version of a role hierarchy, we assign users to groups, assign groups to roles, and then roles to permissions, which leads to a four-tier representation. One advantage of such a representation is that it may reduce the size of the representation more than a three-tier structure.

In order to compute a four-tier, edge-reduced representation of the given bipartite graph G that models the user-permission relation, we proceed as follows:

Initialize Find the role set R and the edges E_{UR} and E_{RP} in an edge-reduced three-tier representation of G, using the methods of Section 5.1.

Iterate while any further reduction is obtained, repeat:

Update groups Find the new set H of groups and edge sets E_{UH} and E_{HR} by computing an edge-reduced three-tier representation of the bipartite user-role relation $G_{UR} \equiv (U \cup R, E_{UR})$; Re-define the group-permission edge set E_{HP} through the roles: $E_{HP} = \{(h,p) \in H \times P \mid \exists\, r \in R,\ (h,r) \in E_{HR},\ (r,p) \in E_{RP}\}$.

Update roles Find the new set R of roles and edge sets E_{HR} and E_{RP} by computing an edge-reduced three-tier representation of the bipartite group-permission relation $G_{HP} \equiv (H \cup P, E_{HP})$; Re-define the user-role edge set E_{UR} through the groups: $E_{UR} = \{(u,r) \in U \times R \mid \exists\, h \in H,\ (u,h) \in E_{UH},\ (h,r) \in E_{HR}\}$.

The process must converge, as it monotonically reduces total edge count. At most three iterations of the while loop were needed on our test cases. Experiments below show that it

Data Set	Users	Permissions	Edges	Role lower bound	Real rank	Maximum Matching
americas large	3,485	10,127	185,294	390	403	688
americas small	3,477	1,587	105,205	172	203	562
apj	2,044	1,164	6,841	453	455	711
emea	35	3,046	7,220	34	34	35
healthcare	46	46	1,486	14	14	46
domino	79	231	730	20	20	21
customer	10,021	277	45,427	276	276	277
firewall 1	365	709	31,951	64	68	242
firewall 2	325	590	36,428	10	10	117

Table 1: Test datasets, lower bounds, real rank

can make a further contribution to reducing the size of the representation.

7. EXPERIMENTAL RESULTS

7.1 Test cases

We applied our algorithms to some network access control rules used in Hewlett Packard (HP) to manage external business partner connectivity. We obtained two user profiles (**americas_small** and **americas_large**) from Cisco firewalls that authenticate external users and provide them with limited HP network access based on their user profiles. We also got similar, smaller datasets **apj** and **emea**. The **healthcare** dataset was obtained from the US Veteran's Administration, which has developed a comprehensive list of the healthcare permissions that may be assigned to licensed or certified providers [20]. The **domino** graph is from a set of user and access profiles for a Lotus Domino server. **customer** is an access control graph obtained from the IT department of an HP customer.

Table 1 gives the sizes of the test case graphs, shows the lower bound computed with the method of Section 4.3.1, and the real rank of the adjacency matrix, which is discussed in Section 4.3.3, and the size of a maximum matching (see Section 4.3.2).

7.1.1 Application of MBC to network reachability analysis

Role discovery is just one application of minimum biclique cover. Others arise whenever a compressed representation of a binary relation is desired. One such relation is reachability in networking. The **firewall{1,2}** datasets are results of running an analysis algorithm on Checkpoint firewalls. These results assert whether packets delivering a service, such as *http*, can reach from some sources to some destinations. Reachability is expressed as set of pairs, each of which is a source IP address range and destination IP address range. The analysis algorithm ensures that the rectangular areas so formed are disjoint. We transformed this set of rectangles into a bipartite graph by partitioning the x-axis and the y-axis into intervals such that each rectangle is a product of intervals, creating a graph vertex for each interval and a graph edge for each rectangle, linking its x-extent (an interval) to its y-extent. A minimum biclique cover of this graph yields a compact description of the set of rectangles.

For example, suppose the three rectangles are $[0,1] \times [0,1]$, $[1,2] \times [3,4]$ and $[0,2] \times [5,6]$. First, we partition the x-axis to $[0,1]$ and $[1,2]$ and the y-axis to $[0,1]$, $[3,4]$ and $[5,6]$. From this we obtain the bipartite graph edges $[0,1] \leftrightarrow [0,1]$ and $[1,2] \leftrightarrow [3,4]$ from the first two rectangles, and $[0,1] \leftrightarrow [5,6]$ and $[1,2] \leftrightarrow [5,6]$ from the third rectangle. This bipartite graph can be covered with two bicliques.

A standard approach to the compaction of a plane region with axis-aligned boundaries is to find a minimum covering by rectangles [9]. In firewall analysis, however, large contiguous regions are infrequent, while cartesian products of unions of intervals, which are bicliques in our bipartite representation, are common. Because of this, we are able to find a very compact representation of the firewall datasets. The representation will be beneficial in several ways, including in the graphical presentation of the information to network administrators.

1**firewall1** 2**firewall2**

7.2 Experimental results for exact cover

We implemented the reduction and coloring of the kernel in C++, and ran it on a PC (Processor: Intel CPU T2500, Processor Speed: 2:00 GHz, RAM : 2 Gb, Compiler: g++/ cygwin, OS: cygwin running on windows). The results are given in Table 2.

Data Set	Reduced Graph |E|	Cover |R|	Cover |E|	Runtime (sec.)
americas large	97	398	87,053	1729
americas small	44	178	9,080	448
apj	0	453	4,256	43
emea	0	34	7,246	.68
healthcare	0	14	194	0.02
domino	0	20	718	0.03
customer	0	276	45,161	72
firewall 1	0	64	1,999	3
firewall 2	0	10	1,158	2

Table 2: Results of exact reduction-based solver

In all cases, we were able to solve the problem exactly. The number of uncovered edges that constitute the irreducible kernel is nonzero only for the two Americas examples, which therefore require a coloring of the complement of the irreducible kernel. The others are completely solved by reduction.

The third column shows the size of the cover found; that is, it shows the biclique cover number of the graph. Thus, we have solved the RMP problem on inputs with as many as 185,000 edges in no more than 30 minutes.

The discovered role sets are small in comparison with $|U|$ and $|P|$ in most cases. For two (**emea** and **customer**), they are not; evidently the **emea** users and the **customer** permissions have almost no exploitable similarities. Note that only in two cases (the Americas) was the lower bound less than the biclique cover number, and in those cases the relative difference is under 3.4%.

In the **americas_large**data set, the largest role that we discovered confers 20 permissions, and 2,804 of the users have this role. About four-fifths of the roles consist of a single user and all of his or her permissions. The discovery of such "orphaned" users and permissions may have significant value in identifying and possibly removing exceptional and in some cases erroneous entries in Access Control Lists.

7.3 Results for heuristic methods

Table 3 shows how well the greedy algorithm for MBC does, with and without the lattice postprocessing, and how well the edge concentration heuristic reduces edges. In four cases (**emea**, **domino**, **customer**, and **firewall2**) we find the optimum cover with the greedy heuristic. We overestimate by about 1% for **apj**, 5% for **americas_large**, 7% for **healthcare**, 11% for 1, and 24% for **americas_small**. The implementations are in Matlab, an interpreted language, and run on a PC with a 3 GHz Xeon processor having 2 GB or DRAM. Runtimes are very low, despite the Matlab implementation; for the large test cases it is 10 to 20 times faster than the reduction / coloring method that finds the optimum.

We applied the edge concentration heuristic of Section 5.1 as well. We started with the greedy role minimizer, then reduced the sum of edges and roles until a local minimum was reached. The results in Table 3 show considerable variability. In some cases, major reductions in edge count (from 75,000 to 22,000 for Americas_large) come at a cost of a modest increase in the number of roles. This happens for the two largest problems. In others, little reduction beyond the contained-role improvement (Section 5.1) can be found.

Since one of the goals of RBAC is to reduce complexity and improve manageability by reducing the size of the access control data structure, we compared models of these sizes. Our model is a count of vertices and edges. Thus, we compare $|E(G)|$, the size of the non-role-based implementation, which is the total number of entries in the set of access control lists. For the latter, we take the number of roles plus the number of edges in the reduced representation generated by our edge concentration heuristic. The data are bimodal: reduction by about half occurs for four of the nine cases, and by an order of magnitude in five of the cases. The reduction is at least 35%, on the large datasets is roughly 90%, and can be as large as 97%.

The rightmost section of Table 3 shows the results obtained by the greedy heuristic followed by the lattice method as a postprocessor to reduce role count. In every case for which the greedy heuristic produces a larger-than-minimum biclique cover, the lattice method reduces the number of roles. This may come at the cost of some increase in the edge count, but there can also be a reduction in edges. Runtime is about the same as the greedy method. The combination of the greedy method with lattice-method postprocessing gets remarkably close to the minimum biclique cover! The difference is at most 12% and on average it is 2%.

The results of the four-tier representation as described in Section 6 are given in Table 4. Note that only modest reductions in complexity are achieved this way.

Data Set	|Groups|	|Roles|	|Edges|	Runtime sec.
americas large	585	983	18,986	505.0
americas small	250	254	7,005	32.0
apj	468	462	4,314	23.0
emea	53	118	3,661	2.1
healthcare	15	14	160	0.1
domino	25	25	415	0.1
customer	585	983	22,860	411.0
firewall 1	77	74	1,472	2.3
firewall 2	11	11	963	0.3

Table 4: Sizes of computed quadripartite representations.

8. CONCLUSIONS

The problem we considered here is to compress a relation via one or a few uses of an "inverse join" construction, and, as such, our approach has potentially very broad applications. It may prove useful to compress and illuminate the structure of important binary relations, and in other applications of the MBC problem.

In the RBAC context, we have shown that instances of the RMP that arise in practice are amenable to exact solution in acceptable runtimes. Moreover, we have presented fast approximate methods that come remarkably close to the exact solutions to the RMP, and that are also effective for the edge concentration problem.

Whether or not this will be important in RBAC is another matter. We focus on the problem of finding the smallest set of roles, or roles and edges, to describe a given set of access controls. However, we have ignored the qualitative but important question of whether or not these roles are meaningful. Indeed, this is the biggest barrier we have encountered to getting the results of role mining to be used in practice; customers are unwilling to deploy roles that they can't understand. In practice, role mining alone is not sufficient. Rather role mining can be viewed as a tool that, along with top-down role engineering, can help make the role engineering process more efficient.

We have also found that in some situations the number of roles we discover is not significantly different from the number of users or the number of permissions. In such cases, role discovery does little to help ease the access management problem. Indeed, it only adds a layer of complexity. However it helps by raising a natural question: why are the existing access controls so complex to begin with? Is this some inherent property of the organization? Are the existing access controls corrupted, or have they been so poorly managed that the discovered roles are fragmented into more roles than necessary to describe the intent of the organiza-

Data	Greedy			Edge Concentration			Lattice		
Set	|R|	|E|	sec.	|R|	|E|	sec.	|R|	|E|	sec.
americas large	422	75,007	69.8	928	21,987	177	400	89,757	50.0
americas small	220	8,730	10.2	258	8,071	13.1	193	8,996	12.0
apj	456	4,182	11.1	471	3,959	12.0	454	4,094	10.9
emea	34	7,246	0.04	104	3,772	0.74	34	7,246	0.03
healthcare	15	211	0.05	15	211	0.05	14	204	0.08
domino	20	729	0.01	28	394	0.05	20	713	0.05
customer	276	45,059	5.93	1,039	23,435	94.1	276	45,047	2.47
firewall 1	70	2,225	0.76	75	1,873	1.02	66	2,008	0.94
firewall 2	10	1,076	0.12	10	1,076	0.16	10	1,091	0.04

Table 3: Heuristic Algorithms: Number of roles and edges (user-role and user-permission connections) for the greedy heuristic (Sect. 4.2.1) alone, after edge concentration (Sect. 5), and after lattice postprocessing (Sect. 4.2.2).

tion? If so, might some approximation technique be a more appropriate approach? We see it as a promising research direction to develop algorithms that elucidate the tradeoff between the complexity of a relation's approximate representation and the closeness of the approximation to the given relation.

After an RBAC structure has been put in place, there will be advantages to periodically optimizing the RBAC structure, due to the constant evolution of organizations. Role mining can assist in this optimization as well.

Acknowledgement We thank Mehul Shah for very helpful discussions on technical content, and the referees, who provided excellent suggestions as to its presentation.

9. REFERENCES

[1] J.A. Bondy and U.S.R. Murty. *Graph Theory with Applications.* North Holland, 1976.

[2] D. Cornaz and J. Fonlupt. Chromatic characterization of biclique covers. *Discrete Mathematics*, 306(5):495–507, 2006.

[3] Edward J. Coyne. Role engineering. In *RBAC '95: Proceedings of the first ACM Workshop on Role-based access control*, page 4. ACM, 1996.

[4] Alina Ene. *Biclique Covers of Bipartite Graphs The Minimum Biclique Cover and Edge Concentration Problems.* 2007. Princeton University.

[5] M.P. Gallagher, A. O'Connor, and B. Kropp. The economic impact of role-based access control. Technical Report Planning Report 02-1, National Institute of Standards and Technology, March 2002.

[6] Floris Geerts, Bart Goethals, and Taneli Mielikäinen. Tiling databases. In *Discovery Science*, volume 3245 of *Lecture Notes in Computer Science*, pages 278–289. Springer-Verlag, 2004.

[7] John E. Hopcroft and Richard M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.

[8] M. Kuhlmann, D. Shohat, and G. Schimpf. Role mining - revealing business roles for security administration using data mining technology. In *SACMAT '03*, pages 179–186. ACM Press, 2003.

[9] V.S. Anil Kumar and H. Ramesh. Covering rectilinear polygons with axis-parallel rectangles. *SIAM Journal on Computing*, 32(6):1509–1541, 2003.

[10] X. Lin. On the computational complexity of edge concentration. *Discrete Applied Mathematics*, 101(1):197–205, 2000.

[11] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *JACM*, 14(5):960–981, 1994.

[12] A. Mehrotra and M.A. Trick. A column generation approach for graph coloring. *INFORMS Journal on Computing*, 8(4):344–354, 1996.

[13] H. Muller. Alternating cycle-free matchings. *Order*, 7(1):11–21, 1990.

[14] J.B. Orlin. Contentment in graph theory: covering graphs with cliques. *Indagationes Mathematicae*, 39:406–424, 1977.

[15] R. Peeters. The maximum edge biclique is NP-complete. *Discrete Applied Mathematics*, 131(3):651–654, 2003.

[16] R. Rymon. Method and apparatus for role grouping by shared resource utilization. U.S. Patent Application 20030172161, September 2003.

[17] J. Schlegelmilch and U. Steffens. Role mining with ORCA. In *SACMAT '05*, pages 168–176. ACM Press, 2005.

[18] Daluss J. Siewert. *Biclique covers and partitions of bipartite graphs and digraphs and related matrix ranks of* $\{0, 1\}$ *matrices.* PhD thesis, The University of Colorado at Denver, 2000.

[19] H.U. Simon. On approximate solutions for combinatorial optimization problems. *SIAM J. Disc. Math.*, 3(2):294–310, 1990.

[20] U.S. Department of Veteran's Affairs. *Licensed Providers Permission Table.* `http://www.va.gov/rbac/docs/20050120PermissionTablesLicensedProviders.doc`.

[21] J. Vaidya, V. Atluri, and Q. Guo. The role mining problem: Finding a minimal descriptive set of roles. In *SACMAT '07*, pages 175–184. ACM Press, 2007.

[22] J. Vaidya, V. Atluri, and J. Warner. Roleminer: Mining roles using subset enumeration. In *ACM CCS '06*, pages 144–153. ACM Press, 2006.

[23] D. Zhang, K. Ramamohanarao, and T. Ebringer. Role engineering using graph optimisation. In *SACMAT '07*, pages 139–144. ACM Press, 2007.